An Interactive Interface for Lighting-by-Example

Hai Nam Ha, Christophe Lino, Marc Christie, Patrick Olivier, van Thuong Nguyen

An Interactive Interface for Lighting-by-Example

Hai Nam HA[1], Christophe Lino[2], Marc Christie[2],
Patrick Olivier[3] and Van Thuong Nguyen[4]

[1] Posts & Telecom. Institute of Technology, Hanoi, Vietnam
[2] IRISA/INRIA Rennes
[3] Newcastle University, UK
[4] VNPT-Hoabinh , Vietnam

Abstract. Lighting design in computer graphics is essentially not a random process but one driven by both a technical and aesthetic appreciation of lighting. In some applications, the result of the lighting design process is a 2D image derived by rendering a 3D scene. Users with limited understandings of manipulation of lighting parameters may have difficulties in properly modifying the lighting parameters in order to achieve desired lighting effects. We present and demonstrate an approach to lighting design in applications where the expected result of the lighting design process is a 2D image. In this approach, the lighting-by-example method using perception-based objective function is used in combination with an interactive interface in order to optimize lighting parameters for an object or a group of objects individually, and the visual results of these separate processes are combined (utilizing 3D depth information) in the seamless generation of a final 2D image.

Keywords: Computer graphics, lighting design, interactive interface, lighting-by-example

1 Introduction

Conventional lighting design is a repeated process of manipulating lighting parameters and testing the changes in lighting effects, and the lighting process stops when desired lighting goals have been achieved. Apparently, conventional lighting design is a knowledge-based process rather than a random process as experienced users would know how to adjust light parameters better than inexperienced users. Manipulating scene parameters, in general, and lighting parameters in lighting design process, in particular, requires heavy interactions between users and graphic tools. Hence, interactions in graphic tools have been continually improved in order to equip users with convenient ways of interactions [1, 6, 7, 8, 9].

In graphic tools such as 3D Studio Max, lighting work starts with identifying the types of light sources and their characteristics that will be used in a scene. By investigating the purpose and intent of lights, a user finds a real-world counterpart for each of every light used in the scene. Lights are then positioned at intended locations. The next step will be editing the properties of the lights. In commercial graphic tools, most of the light properties such as intensities, colour, attenuation and shadow parameters are manipulated through a window using keyboard. For some properties of lights such as light position and direction, there is an alternative for specifying them through mouse-based interactions such as dragging and dropping. Indeed, mouse-based interactions for light positioning would be more efficient and intuitive than specifying the values through a window using keyboard as users normally know relatively where the light should be and which direction the light should point at rather than the exact values for light position and direction. In reality, commercial graphic tools such as 3D Studio Max, Maya, Light Wave support mouse-based interactions for manipulating light position and direction. Where mouse-based interactions cannot be easily applied such as specifying values for falloff and hotspot of lights, context menus are normally used in graphic tools to enhance the performance of accessibility to functionalities.

The ultimate goal of research on human-machine interactions is to empower users with convenient ways of achieving the desired goals with less effort using limited resources. Particularly, interactions in lighting design aims to provide users with convenient ways of achieving desired lighting effects by taking advantages of the combination of existing lighting design approaches.

2 Interactive Design Through Independent Object Lighting

Interactive lighting design is problematic, as although we may have a clear notion of how we want each individual object in a scene to be lit, any change in lighting parameters effects the illumination of all objects simultaneously. One solution, which is already a common practice in graphic design, is to produce 2D images from 3D scenes and use 2D image processing techniques to merge different source images (i.e. source images that have been differently lit or have been manipulated in some way). There are a number of drawbacks to working with 2D images; in particular, that 3D information is not taken into consideration. Indeed, when 2D images are merged together without taking advantages of 3D information, unanticipated effects may occur (for example, if no account is taken of object occlusion).

In fact, creating lighting effects, either using 2D compositing, or by modifying the parameters of scene lights directly, is not an intuitive process, even for experienced graphic artists. To address this problem, we propose the extension of the work presented so far. By integrating the optimization schemes developed in [2,3,4] (for both *perception-based lighting* and *lighting-by-example*) we have developed an interface that allows users to design the scene lighting on an object-by-object basis. Developing an interactive approach requires us to extend of our optimization framework and develop mechanisms by which we can more or less independently specify and modify the lighting for different objects in a scene.

Two methods are proposed which can be contrasted in two respects, the degree to which they treat each object in the scene independently, and the nature of the result of the lighting process. Both approaches take as a starting point (if desired) either the results of a perception-based lighting optimization process, or a lighting-by-example process. In this way we envisage *interactive lighting design* as a matter of fine tuning the illumination of each object in a scene. In the first approach, 3D Interactive Lighting Design (3D-ILD), the outcome of the lighting design process is a 3D scene in which objects are differently lit using spotlights, as is commonly used in interactive graphics applications. In the second approach (2D-ILD) a set of lights that is specific to each object is optimized, and the visual results of these separate processes are combined (utilizing 3D depth information) in the seamless generation of a 2D image.

2.1 3D Interactive Lighting Design (3D-ILD)

For most graphics applications, we anticipate that the result of a lighting design system is a configuration of lights with respect to the scene elements. Though this design process takes place by reference to a single viewpoint (that is, the optimization process) the viewpoint selected is typically one that is characteristic or exemplifies the contexts in which an object is likely to be viewed. Thus the resulting lighting configuration can be integrated into a scene which includes other objects and light. We have realized this 3D interactive lighting design process through the use of spotlights which are used to light each object individually. A spotlight provides localized and directional illumination. Illumination is restricted to within a specified cone (i.e. the beam of light). This beam of light can be controlled through the specification of both width of cone and the nature of the drop off function

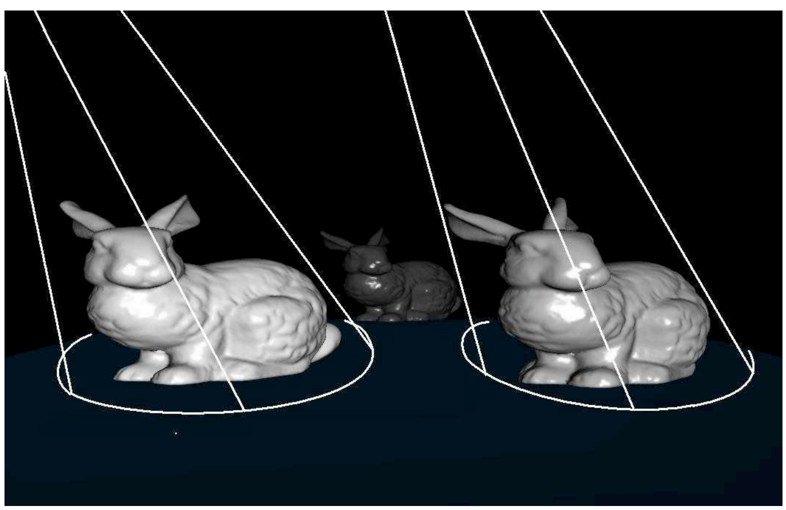

Figure 1. Object is lit by a spotlight with an appropriate focus cone.

Figure 1 illustrates the idea of using spotlights to optimize lighting for objects separately. In this scenario, there are 3 objects in the scene and for simplicity 2 spotlights are used to light the leftmost and rightmost objects, and an ambient light was used to create the background illumination. The cone around each object represents the focus of a spotlight. Apparently, the cone parameters must be adjusted such that whole object is in the cone of the spotlight.

The first step of the interactive design process involves setting up appropriate parameters for the spotlight, the position, orientation, and cut-off angle, such that the cone of the spotlight focuses on the object of interest without lighting adjacent objects (see figure 2(a)). This is sometimes impossible due to the close proximity of other object, and occlusions, in the 3D scene. In such cases the user simply has to adjust the light so as to keep lighting on unintended objects to a minimum. Once the parameters of spotlights for different objects have been set up, the next step is to select the spotlight to be optimized (see figure 2(b)).

The final step is to choose the optimization technique to be employed. This can be either ideal *perception-based lighting optimization* [3,4] which will optimize the lighting parameters of the spotlight such that the visual properties of the objects in the scene are maximized; or *perception-based lighting-by-example* [5] which aims to capture the lighting effect of a 3D example and recreate it for the current object (see figure 2(c)). Our current experimental implementation supports up to 6 spotlights. Figure 3 shows an example in which lighting for objects are separately optimized using spotlights. Figure 3(a) shows the original scene for which there is a default direction lighting originating from behind and above the camera. Figure 3(b) shows the result of applying the ideal perception-based lighting optimization approach to the left-most bunny rabbit. Figure 3(c) shows the example object (an elephant) that is the target for lighting the rightmost bunny rabbit using perception-based lighting by example. Finally, figure 3(d) shows the final result for both processes. The figures demonstrate how the objects can be independently lit in a 3D scene without significantly affecting the lighting of the other objects in the scene.

The optimization steps of the interaction took 15 steps (93 seconds on a Windows PC, with P4 3.00GHz processor, 1G RAM and GeForce 7600 GT graphics card) for the ideal perception-based lighting optimization and 28 steps (154 seconds on a Windows PC, with P4 3.00GHz processor, 1G RAM and GeForce 7600 GT graphics card) for the perception-based *lighting-by-example* optimization.

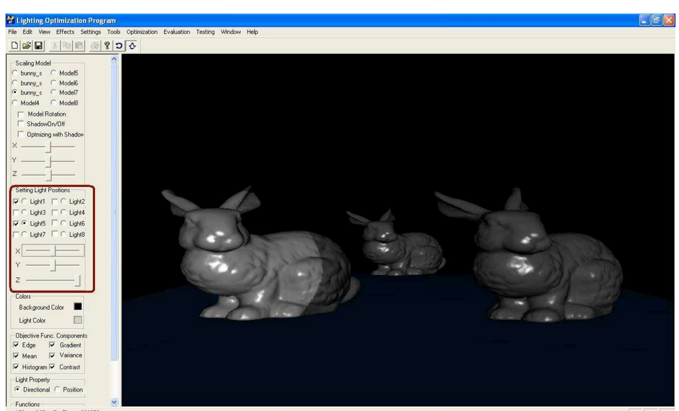

Figure 2. The leftmost object is not completely in the cone of the spotlight.

2.2 2D Interactive Lighting Design (2D-ILD)

Where the result of the lighting design process is a 2D image (for example, in graphic design for static digital media and print) we can design the lighting for each object in the scene by optimizing each object individually. Figure 4 illustrates this process, by which the other objects in a scene are hidden and the techniques that we developed in [3,4,5] are deployed to optimize the lighting for a single object. Having optimized each object separately, the resulting 3D scenes are merged into a final (2D) composition using depth information which is retained from the 3D renderings. The final image buffer is displayed in a separate window.

Figure 5 is a diagram that illustrates the use of interactive interface to individually optimize lighting parameters for objects. Firstly the objects of the 3D scene are loaded and the graphic designer then uses a context menu to hide the objects that are not to be optimized in current loop. The objects for which lighting parameters are to be optimized in this loop remain visible. At this point the user can also manipulate the visible objects with operations such as: rotating, scaling and moving the objects. When the user is satisfied with current set-up of the objects, he can proceed the next step of choosing an appropriate lighting optimization, either ideal perception-based lighting, perception-based *lighting-by-example* or wavelet-based *lighting-by-example*. The optimization for the current setup of objects proceeds and the colour buffer is then merged to the final image buffer. The user reveals all the invisible objects, and if lighting parameters have been optimized for all objects and the optimization process stops. If not, the user will start to optimize lighting parameters for the next object and new objects may also be loaded to the scene.

Figure 6 shows a worked example in which the lighting for each of the faces on a two-headed bust is separately designed and then recombined into a final image. In this process a context menu for an object provides a number of options:

1. *Hide This Object*: hide the currently selected object, to allow the optimization of the remaining (visible) objects in the scene.
2. *Hide All Objects*: hide all objects of the 3D scene, used when a user is about to load a new object to the scene and lighting parameters are only to be optimized for the newly loaded object.
3. *Show All Objects*: show all objects of the 3D scene, used when the user wants to view the set-up of the whole 3D scene.
4. *Merge With Final Image:* merges the colour buffer with the final image buffer using available depth information to ensure that objects of the two buffers are merged in the right depth order.

Figure 6(a) shows the selection of the right head as the object to hide, in figure 6(b) the ideal perception based framework is deployed, and in figure 6(c) the colour and depth information is merged in the final 2D image (and a separate depth buffer). Figure 6(d) shows the selection of the left head as the object to hide. As the perception-based *lighting-by-example* framework is deployed, figure 6(e) shows the exemplar, and figure 6(f) shows the result for the right head. Figure 6(g) shows the final merged image in which the spatial information for the scene is preserved but the two heads have been lit very differently.

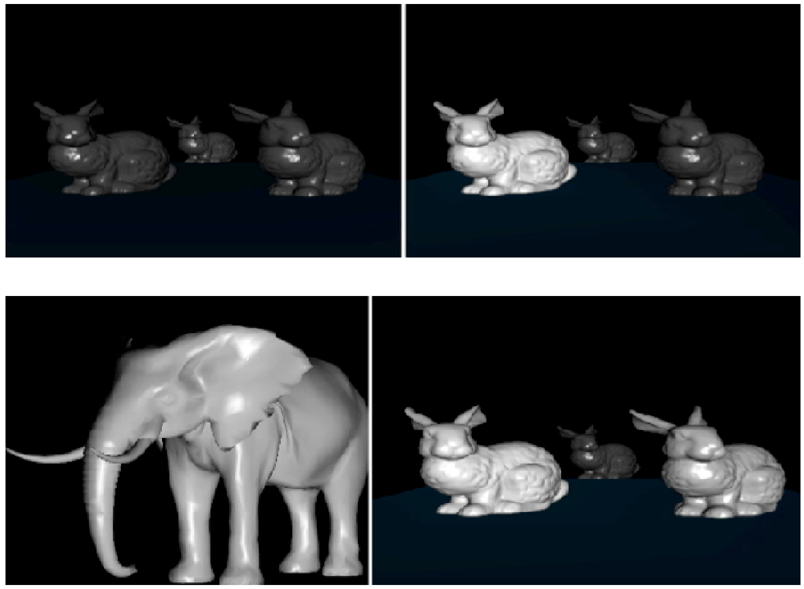

Figure 3. Lighting parameters separately optimized using spotlights: original set-up (top-left); applied to left-most bunny only (top-right); target for right-most bunny (bottom-left); and final result after both processes (bottom-right).

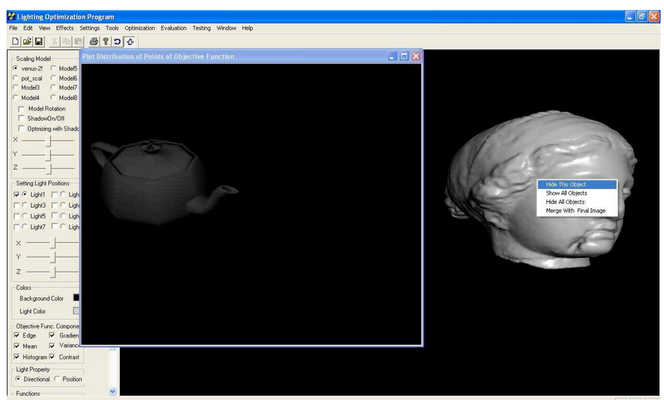

Figure 4. Interactive interface of the lighting design application.

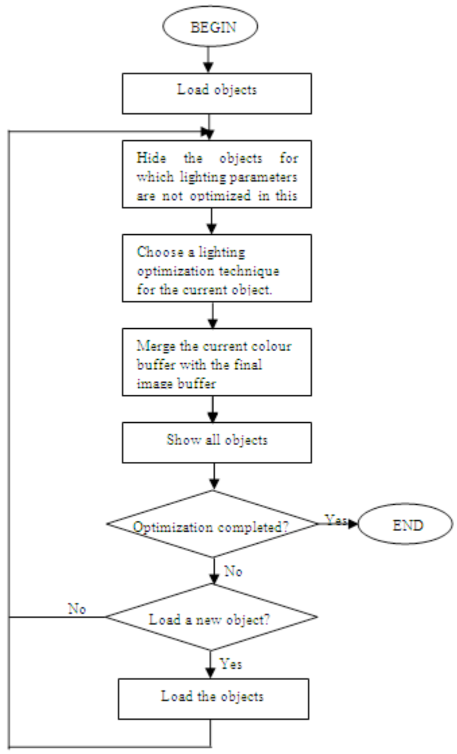

Figure 5. The workflow for optimizing lighting parameters for an individual object in 2D Interactive Lighting Design.

3 Conclusion

In this paper we have explored different ways of using lighting design methods. The use of spotlights in 3D context where the outcome is a 3D scene with optimized lighting parameters has been explored in which each object in the scene was lit by a spotlight that can be individually optimized. The challenge of this approach is that the parameters of the spotlights must be carefully adjusted such that each spotlight can light only one object. Interactive methods developed in this paper aim to demonstrate some ways of using lighting design approaches developed in this research. There should be more different ways of using lighting design approaches developed in this research.

In the current interactive interface, we only implemented a number of basic function as for the purpose of exploring their potential, and as a result our prototype interface was rather simple in the range of interactions it supports for the spotlights and the depth information-supported merging technique. We believe the further development of interaction techniques, to combine the lighting approaches, is likely to greatly enhance the appeal of such a system to novice and expert users. For example, an interface that allows users to change the target values for *lighting-by-example* approach would allow significantly more flexibility. Likewise, the interface for manipulation of light parameters is rather limited in current interactive interface. In many cases, the light positions have been fixed and defined by users. A sketch-based interface for manipulating scene parameters would also contribute to making interactions far more natural for users. Also, in some applications, designing lighting parameters for multiple viewpoints is highly desirable, and this requires interaction and display techniques such as allowing multiple viewports for displaying results of the lighting design process from different viewpoints. Finally, in the current development framework, only standard lights are used in the system. However, for scenes require complex lighting effects, extended light sources (i.e. physically extended light sources that are not point-like) are needed.

Acknowledgements

This work was supported in part by the EU FP7 network of excellence IRIS (ICT-231824) Integrating Research in Interactive Storytelling.

References

1. A Lighting Approach for Computer Graphics (SIGGRAPH 96 - Course 30)
2. Shacked, R., Lischinski, D.: Automatic Lighting Design using a perceptual quality metric. In Proc. Eurographics '01, vol. 20, no.3, 2001.
3. Ha, H.N., Olivier, P.: Perception-based lighting design. In: Theory and Practice of Computer Graphics, Middlesbrough, UK, 2006.
4. Ha, H.N., Olivier, P.: Explorations in Declarative Lighting Design. In: 6th International Symposium on Smart Graphics. Vancouver, Canada, 2006.
5. Ha, H.N., Olivier, P.:Perception-based lighting-by-example. In: Theory and Practice of Computer Graphics (TPCG), 2007.
6. Rusinkiewicz, S., Burns, M., DeCarlo, D.: Exaggerated Shading for Depicting Shape and Detail. ACM Transactions on Graphics (Proc. SIGGRAPH). 25(3) July, 2006.
7. John, K. K., Painter, J.S., Cohen, M.F.: Radioptimization: goal based rendering. In: Proceedings of SIGGRAPH'93. pp.147~154, 1993
8. Jung, T., Gross, M. D., Do, E. Y.-L.: Light pen – sketching light in 3d. In: CAAD Futures, 2003.
9. Okabe, M., Matsushita, Shen, Y. L., Igarash, T.: Illumination Brush: Interactive Design of All-frequency Lighting In: Pacific Graphics, 2007.

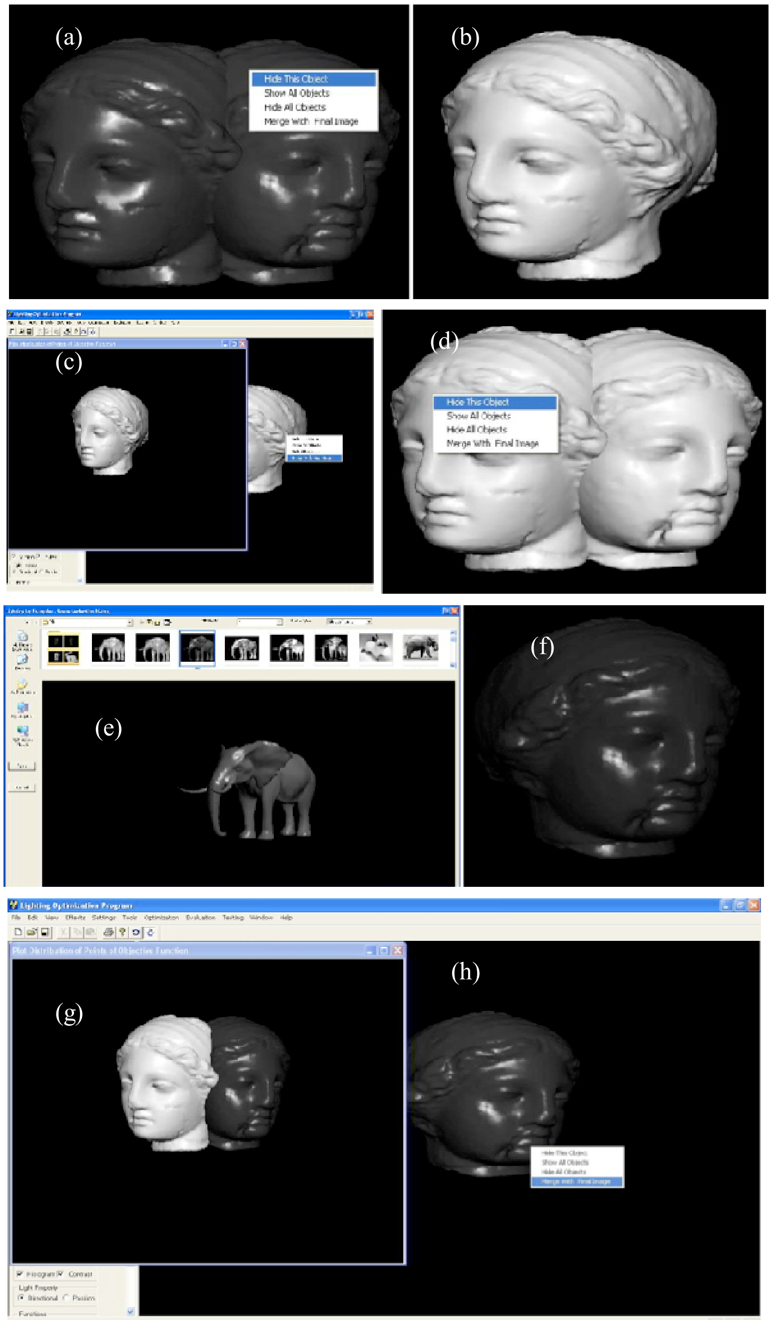

Figure 6. Lighting two heads differently using different lighting design.